BeSAT: Behavioral SAT-based Attack on Cyclic Logic Encryption

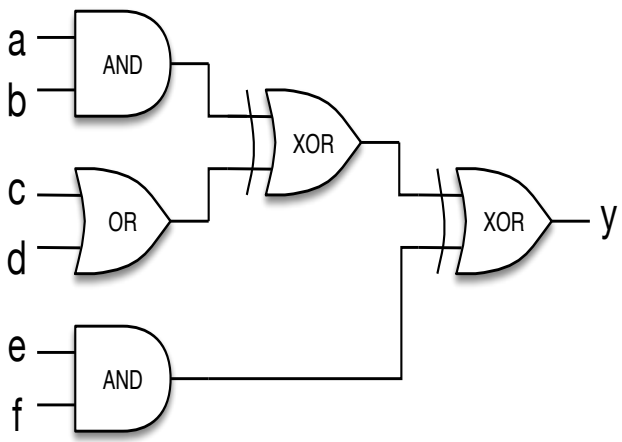
Yuanqi Shen, You Li, Amin Rezaei, Shuyu Kong,
David Dlott, Hai Zhou
EECS Department at Northwestern University

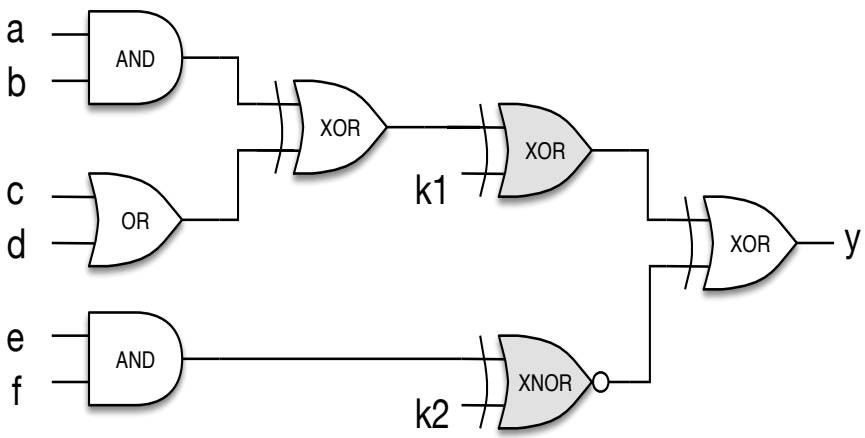
Threats of Outsourcing

- Design houses outsourced their fabrication to offshore foundries for low cost
- Many offshore foundries are hard to be trusted
  - Counterfeiting, piracy, unauthorized overproduction, etc.
- Loss: billions per month

Logic Encryption

- Proposed to thwart counterfeiting, piracy, and unauthorized overproduction
- Insert extra gates into IC design to hide its original functionality
- Many years research

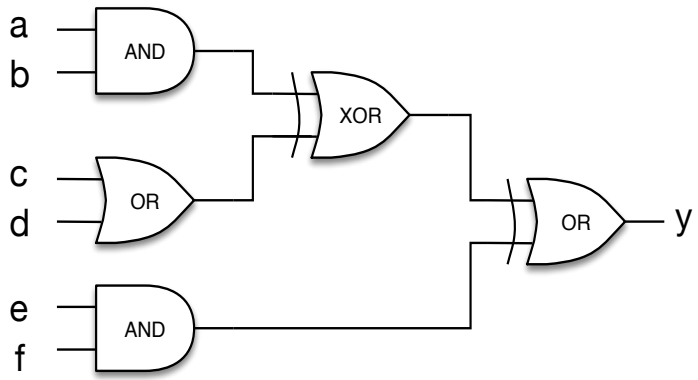


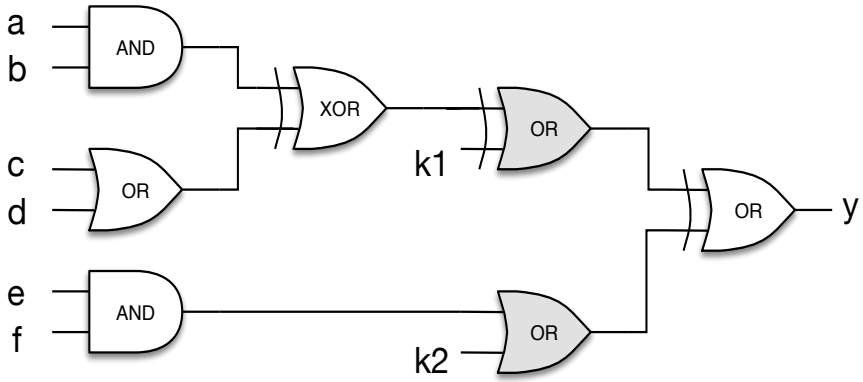


SAT Attack

- A logic decryption algorithm that has successfully corrupted all existing logic encryption approaches up to 2015
- **Idea**: use SAT solver to iteratively find Distinguishing Input Patterns (DIPs) and their correct outputs to prune out wrong keys
  - $C(X, K_1, Y_1) \wedge C(X, K_2, Y_2) \wedge (Y_1 \neq Y_2)$
- Only need a small number of DIPs to exclude all wrong keys

Example

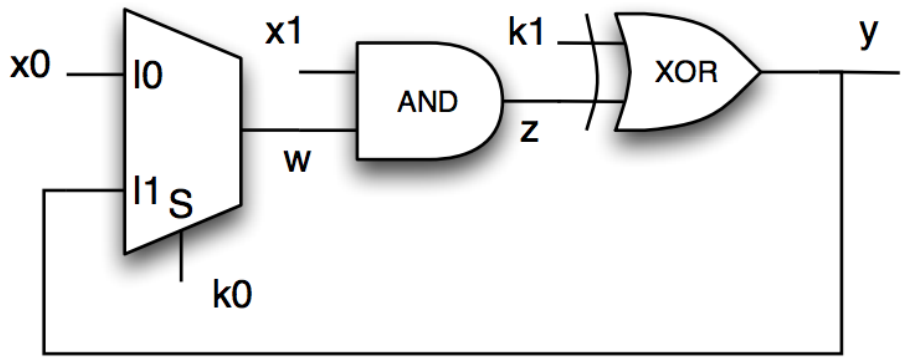




- First iteration:
  - $(x_1, y_1) = (000000, 0)$, $(k_1, k_2) \neq \{(1,0), (1,1), (0,1)\}$
- Second iteration:
  - UNSAT, $(k_1, k_2) = (0,0)$

Cyclic Logic Encryption

- Cyclic logic encryption is proposed by adding cycles to the encryption scheme

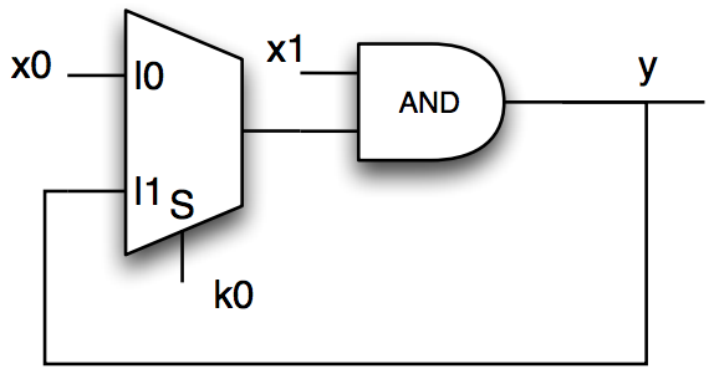


- Original circuit is an NAND gate. Correct key is $(k_0, k_1) = (0, 1)$
- Introducing statefulness and oscillation
- Problems for SAT attack!

Statefulness

Definition

Assume $c_1(x, k)$ and $c_2(x, k)$ are two copies of an encrypted circuit. Statefulness indicates $\exists$ an input $\hat{x}$ and a key $\hat{k}$, s.t. $c_1(\hat{x}, \hat{k}) \neq c_2(\hat{x}, \hat{k})$.



- When $k_0 = 1$ and $x_1 = 1$, y is undetermined
- Constraint $C(X, K_1, Y_1) \wedge C(X, K_2, Y_2) \wedge (Y_1 \neq Y_2)$ always satisfies!
- Repeated DIPs are found, SAT attack cannot terminate!

Oscillation

Definition

Oscillation indicates $\exists$ *an input* $\hat{x}$ *and a key* $\hat{k}$*, s.t.* $SAT(c(\hat{x}, \hat{k})) =$ *unsatisfiable.*

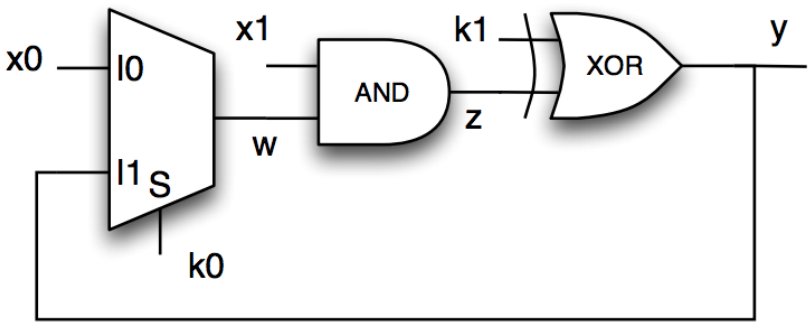


Figure: Encrypted Circuit

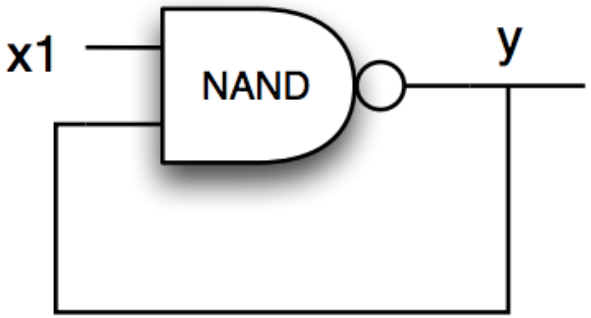


Figure: Circuit when $(k_0, k_1) = (1, 1)$

- When $(k_0, k_1) = (1, 1)$, the circuit becomes an NAND gate with a feedback
- There is always a conflict
- UNSAT, a correct key cannot be returned!

CycSAT

- CycSAT is proposed to incorporate a Conjunctive Normal Form (CNF) formula to the original SAT attack that captures the acyclic assumption.
  - Structure analysis

Example

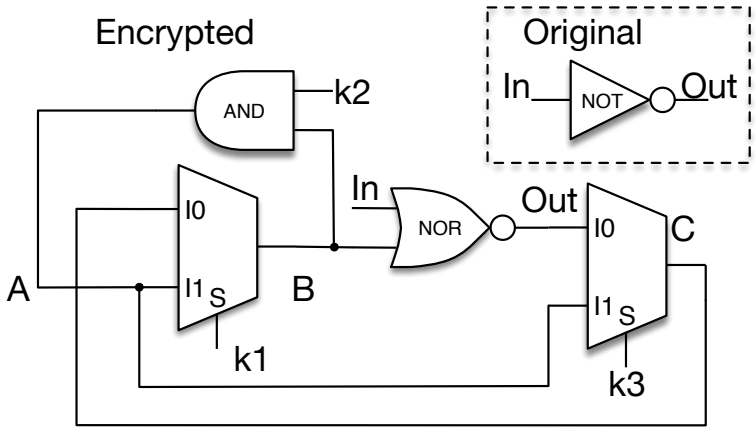


Figure: Encrypted Circuit

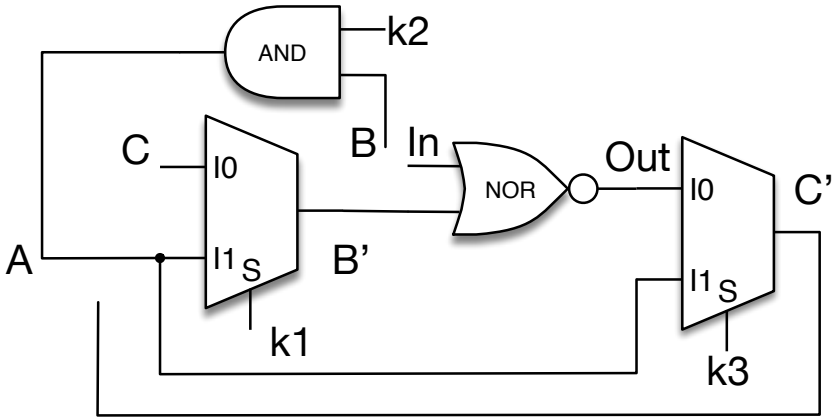


Figure: Feedback Broken

- Topological order: (A, B, Out, C)
- B and C are feedback wires and broken in CycSAT

$$F(B, B') = F(B, A) \vee F(A, B') = \neg k_2 \vee \neg k_1$$
$$F(C, C') = F(C, B') \vee F(Out, C') = k_1 \vee k_3$$
$$NC = F(B, B') \wedge F(C, C') = (\neg k_2 \vee \neg k_1) \wedge (k_1 \vee k_3)$$

Example

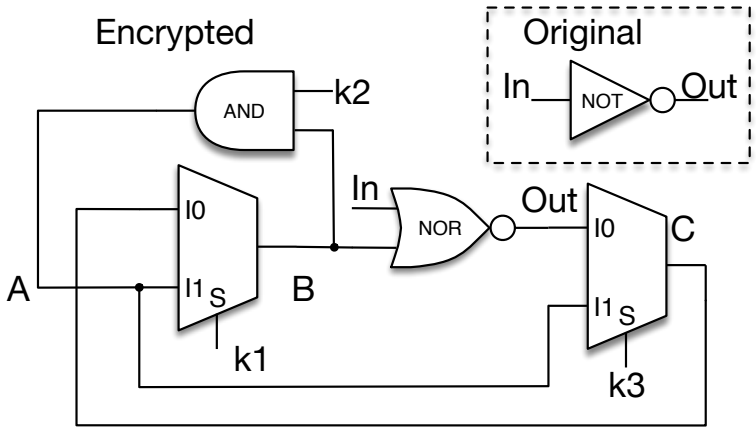


Figure: Encrypted Circuit

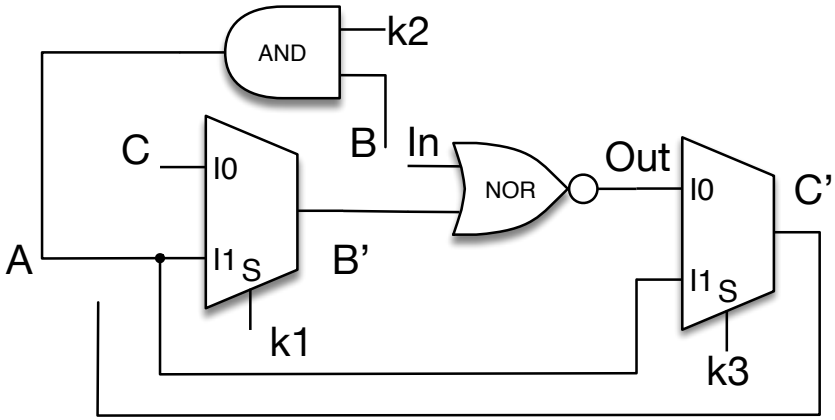


Figure: Feedback Broken

$$F(B, B') = F(B, A) \vee F(A, B') = \neg k_2 \vee \neg k_1$$
$$F(C, C') = F(C, B') \vee F(Out, C') = k_1 \vee k_3$$
$$NC = F(B, B') \wedge F(C, C') = (\neg k_2 \vee \neg k_1) \wedge (k_1 \vee k_3)$$

- $k_1 = 0, k_2 = 1, k_3 = 1$
- cycle C, B, A, C with feedback wires B and C still exists!

Theorem

Theorem

The edges in a directed graph cannot always be divided into two disjoint sets, such that any simple cycle is formed by two simple paths, one composed of edges from each set

- If more than one backward edges are involved in a cycle, they possibly do not form a simple path on this cycle
- The traversal in any topological order of vertices ignoring these backward edges does miss this cycle
- CycSAT is impossible to efficiently capture the condition to break all possible cycles
- **Enhancement**: introduce behavior analysis

Statefulness

Theorem

If a repeated DIP is found in an iteration of CycSAT, at least one of the keys found by the SAT solver in this iteration causes statefulness.

- **Solution**: If repeated DIPs happen in CycSAT, directly block the stateful key
  - Evaluating the correct output under the repeated DIP
  - Comparing outputs with the correct output
  - A stateful key if outputs are different

Oscillation

- **Solution**: the ternary simulation combined with the Boolean satisfiability problem
- A technique to check if the circuit is combinational
- It is arguably stringent, but oscillation is unlikely to happen in real design

Theorem

If the ternary-based SAT method returns unsatisfiable on a circuit $c(x, k)$, $\nexists$ *an input* $\hat{x}$ *and a key* $\hat{k}$ *s.t.* $SAT(c(\hat{x}, \hat{k})) =$ *unsatisfiable.*

- If the ternary-based SAT method returns satisfiable, the circuit is non-combinational and the key should be pruned

BeSAT Algorithm

- Generating the "no structural path" formula, and adding its CNF formula into the existing constraint
- Running SAT attack with cyclic analysis
  - If repeated DIPs are found, directly blocking stateful keys
  - When the SAT solver cannot find any DIPs, using the ternary-based SAT method to verify the case of oscillation

Theorem

When BeSAT terminates and returns a key value, the returned key is guaranteed to be correct.

Evaluation

Cyclic encryption with 10 intentional feedbacks of cycle lengths 10

	cyclic encrypted circuits					CycSAT		SAT−based attack		BeSAT	
circuit	#primary inputs	#key inputs	#outputs	#gates	#cycles	key	#iterations	key	#iterations	key	#iterations
apex2	39	20	3	600	20	yes	4	no	–	yes	4
apex4	10	20	19	5380	20	yes	3	no	–	yes	3
c432	36	20	7	180	20	no	–	no	–	yes	17
c499	41	20	32	222	20	yes	3	yes	8	yes	4
c880	60	20	26	403	20	no	–	no	–	yes	19
c1355	41	20	32	566	20	yes	25	no	–	yes	28
c1908	33	20	25	900	20	yes	34	yes	167	yes	8
c2670	233	20	140	140	20	no	–	no	–	yes	32
c3540	50	20	22	1689	20	no	–	no	–	yes	31
c5315	178	20	123	2327	20	no	–	yes	15	yes	14
c7552	207	20	108	3532	20	no	–	no	–	yes	5
dalu	75	20	16	2318	20	yes	16	yes	9	yes	12
des	256	20	245	6493	20	yes	5	no	–	yes	5
ex5	8	20	63	1075	20	yes	11	no	–	yes	11
ex1010	10	20	10	5086	20	yes	3	no	–	yes	3
i4	192	20	6	358	20	yes	3	no	–	yes	3
i7	199	20	67	1335	20	yes	7	yes	6	yes	7
i8	133	20	81	2484	20	yes	8	yes	7	yes	8
i9	88	20	63	1055	20	yes	3	yes	11	yes	4
k2	46	20	45	1835	20	no	–	no	–	yes	37
seq	41	20	35	3559	20	yes	5	no	–	yes	5

Conclusion

- Recently, cyclic logic encryption is proposed as a newly encryption technique to defeat many existing attacks
- Even though CycSAT is proposed to compute the "no cycle" formula, it may miss cycles
- In order to overcome the drawback of CycSAT, we propose BeSAT, which brings in the behavioral analysis
- The experimental result indicates BeSAT can efficiently solve the correct key value of cyclic encrypted benchmarks

Thank you!